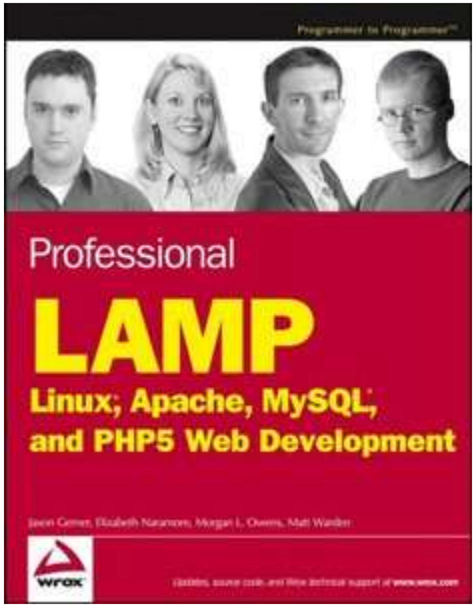
The M in LAMP: MySQL



Programmer to Programmer™

Professional

LAMP

Linux, Apache, MySQL, and PHP5 Web Development

Jason Gerner, Elizabeth Naramore, Morgan L. Owens, Matt Warden

wrox

Updates, source code, and Wrox technical support at www.wrox.com

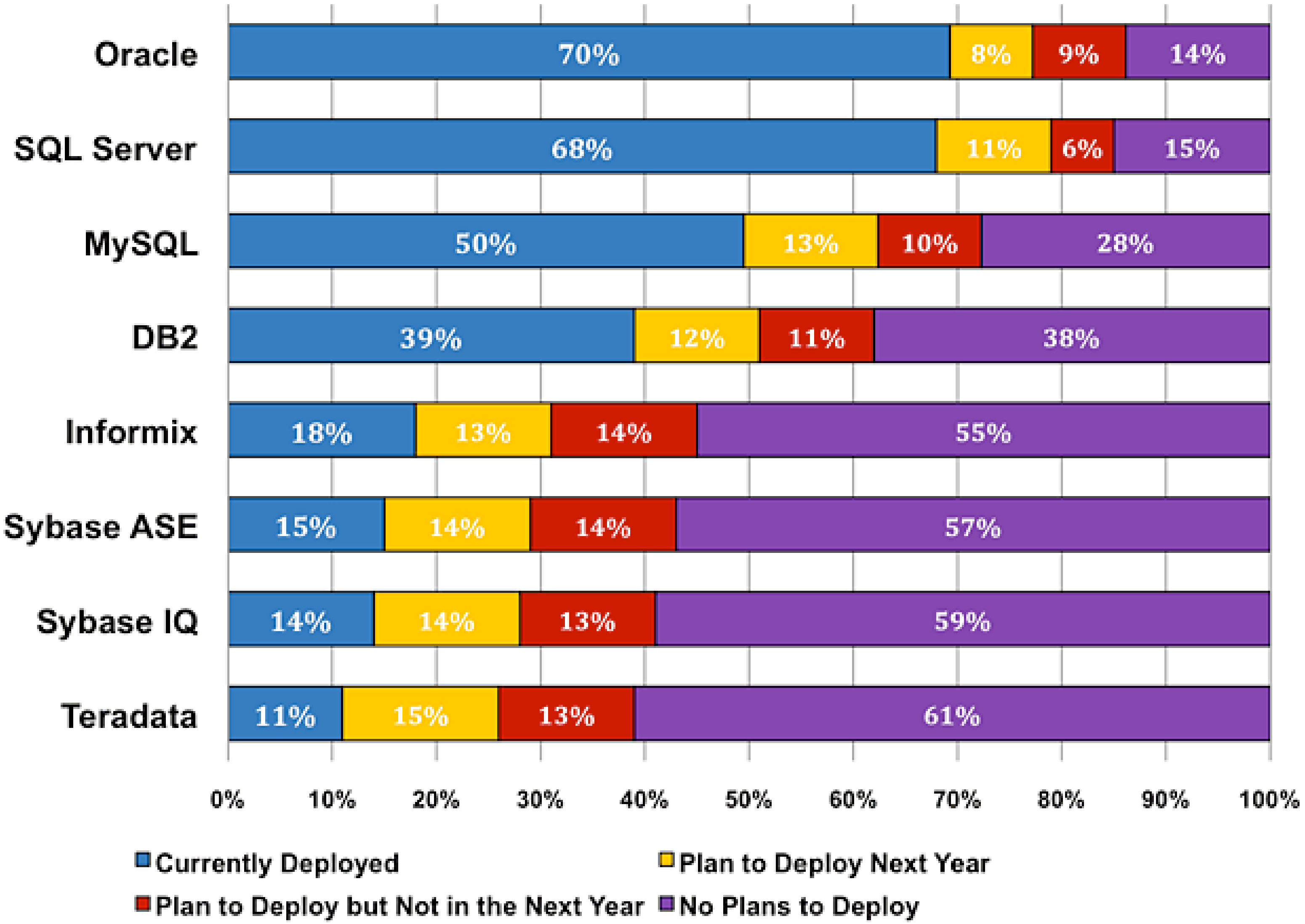
Overview

- MySQL
  - Setup, using console
  - Data types
  - Creating users, databases and tables
- SQL queries
  - INSERT, SELECT, DELETE
  - WHERE, ORDER BY, GROUP BY, LIKE, LIMIT, COUNT(*)
- Using from PHP
  - Procedural vs. Object-oriented
  - Iterating over results

MySQL history

- MySQL
  - "My ess que ell", "My sequel"
  - 1995, MySQL AB founded in Sweden
  - 2000, goes open source
  - 2003, 4 million active installations, 30K downloads/day
  - 2006, 33% market share, 0.2% of revenue
  - 2008, acquired by Sun for $1B

MySQL installations



Some numeric data types

Type	Storage	Signed range
TINYINT	1 byte	-128 to +127
SMALLINT	2 bytes	-32768 to +32767
MEDIUMINT	3 bytes	-8388608 to +8388607
INT, INTEGER	4 bytes	-2147483648 to +2147483647
BIGINT	8 bytes	-9223372036854775808 to 9223372036854775807

Type	Storage	
FLOAT	4 bytes	Single precision, approximate
DOUBLE	8 bytes	Double precision, approximate
DECIMAL(x,y)	varies	Exact value, x significant figures, y decimal places
BIT(M)	varies	Stores 1-64 bits

Some string data types

Type	
CHAR(X)	Fixed-length text data, 0-255 in length
VARCHAR(X)	Variable-length text data, 0 to 65,535 in length
BLOB	Binary Large Object, used to store large amounts of binary data such as image or files, 64K max length
TEXT	Large amounts of text data, 64K max length
TINYBLOB TINYTEXT	255 max length
MEDIUMBLOB MEDIUMTEXT	16,777,215 max length
LONGBLOB LONGTEXT	4,294,967,295 max length
ENUM	Enumerated type, string value in a specified set of allowed values

Some date/time data types

Type	
DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS Automatically update when row changed
TIME	HH:HMM:SS
YEAR(M)	YY, or YYYY

Setting up a database

- Log in as root
- Create a new database:

```
CREATE DATABASE grocery;
```

- Create a new user, grant privileges:

```
CREATE USER 'username'@'localhost'
IDENTIFIED BY 'pwd';

GRANT ALL PRIVILEGES ON grocery.*
TO 'username'@'localhost';
```

Creating a table

- Table creation syntax:

```
CREATE TABLE table_name (col_name1 col_type1,
                         col_name2 col_type2, ...)
```

```
CREATE TABLE inven
(
   id      INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
   name    VARCHAR(50) NOT NULL,
   details TEXT,
   price   FLOAT NOT NULL,
   qty     INT NOT NULL
);
```

Inserting data into a table

- Insertion syntax:

```
INSERT INTO table_name (col_name1, col_name2, ...)
                VALUES (col_val1, col_val2, ...);
```

```
INSERT INTO inven (name, details, price, qty)
VALUES ('Apples', 'Ripe apples.', '0.25', 1000);
```

```
INSERT INTO inven (name, details, price, qty)
VALUES ('Apples', 'Rotten apples.', '0.02', 594);
```

```
INSERT INTO inven
VALUES (NULL, 'Apples', 'Ripe apples.', '0.25', 1000);
```

Need to include values for every column if you don't provide column name list!

Selecting data form a table

- Select syntax:

```
SELECT col_name1, col_name2, ... FROM table_name
[WHERE condition]
[GROUP BY col_name]
[ORDER BY condition [ASC | DESC]]
[LIMIT [offset,] rows]
```

```
SELECT *
FROM inven;
```

```
SELECT name, qty
FROM inven;
```

```
SELECT name, qty
FROM inven
ORDER BY price
LIMIT 2;
```

Selecting data form a table

- Select syntax:

```
SELECT col_name1, col_name2, ... FROM table_name
[WHERE condition]
[GROUP BY col_name]
[ORDER BY condition [ASC | DESC]]
[LIMIT [offset,] rows]
```

```
SELECT *
FROM inven
WHERE qty <= 500;
```

```
SELECT *
FROM inven
WHERE name LIKE 'a%';
```

Any names that begin with a.

Selecting data form a table

- Select syntax:

```
SELECT col_name1, col_name2, ... FROM table_name
[WHERE condition]
[GROUP BY col_name]
[ORDER BY condition [ASC | DESC]]
[LIMIT [offset,] rows]
```

```
SELECT *, count(*) as freq
FROM inven
GROUP BY name;
```

Causes generation of a new column that counts number of rows that were aggregated by GROUP BY clause

Deleting data from a table

- Delete syntax:

```
DELETE FROM table_name
[WHERE condition]
[LIMIT rows]
```

```
DELETE FROM inven;
```

```
DELETE FROM inven
WHERE qty < 500;
```

Using MySQL from PHP

- PHP's MySQL extension
  - Original extension
  - mysql_* functions
- PHP's mysqli extension
  - New improved extension
  - Takes advantage of new MySQL v4.1.3+ features
  - Supported in PHP v5+
  - Object-oriented interface
  - Support for multiple statements
  - Support for transactions
  - mysqli_* functions

Procedural style

```
<?php

   $mysqli = mysqli_connect("localhost", "root", "password", "inven");
   if (mysqli_connect_errno())
   {
      printf("Connect failed: %s\n", mysqli_connect_error());
      exit();
   }

   $sql = "SELECT * FROM inven";
   $res = mysqli_query($mysqli, $sql);
   if ($res)
   {
      while ($newArray = mysqli_fetch_array($res, MYSQLI_ASSOC))
      {
          $name     = $newArray['name'];
          $details  = $newArray['details'];
          $price    = $newArray['price'];
          echo "$name $details $price <br />";
      }
      mysqli_free_result($res);
   }
   mysqli_close($mysqli);

?>
```

Object-oriented style

```
<?php

   $mysqli = new mysqli("localhost", "root", "", "grocery");
   if ($mysqli->connect_errno)
   {
      printf("Connect failed: %s\n", $mysqli->connect_error);
      exit();
   }

   $sql = "SELECT * FROM inven";
   $res = $mysqli->query($sql);
   if ($res)
   {
      while ($newArray = $res->fetch_array(MYSQLI_ASSOC))
      {
          $name     = $newArray['name'];
          $details  = $newArray['details'];
          $price    = $newArray['price'];
          echo "$name $details $price <br />";
      }
      $res->close();
   }
   $mysqli->close();

?>
```

Summary

- MySQL
  - The most popular open source database
  - More than good enough for most web apps
  - Supports standard SQL syntax
    - SELECT, INSERT, DELETE, UPDATE
    - WHERE, ORDER BY, GROUP BY, LIMIT
    - LIKE, COUNT
  - Use in PHP
    - Procedural style
    - Object-oriented style